

Training vegetable

2.Image Augmentation

In [1]:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [25]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os
import torch
import torchvision
import tarfile
import torchvision
from torch.utils.data import random_split
from torchvision.datasets import ImageFolder
from torchvision import transforms
from torchvision.transforms import ToTensor
from torch.utils.data.dataloader import DataLoader
import torch.nn as nn
from torchvision.utils import make_grid
import torchvision.models as models
import torch.nn.functional as F
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:

```
train_datagen = ImageDataGenerator(rescale=1./255,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
```

In [3]:

```
test_datagen = ImageDataGenerator(rescale=1./255)
```

In [6]:

```
xtrain =
train_datagen.flow_from_directory('/content/drive/MyDrive/Classroom/Dataset
Plant Disease/Veg-dataset/Veg-dataset/train_set',
                                target_size=(64,64),
                                class_mode='categorical',
                                batch_size=10)
```

Found 10410 images belonging to 9 classes.

In [33]:

```
data_dir='/content/drive/MyDrive/Classroom/Dataset Plant Disease/Veg-
dataset/Veg-dataset/train_set'
```

In [34]:

```
transformer = torchvision.transforms.Compose(
    [ # Applying Augmentation
      torchvision.transforms.Resize((224, 224)),
      torchvision.transforms.RandomHorizontalFlip(p=0.5),
      torchvision.transforms.RandomVerticalFlip(p=0.5),
      torchvision.transforms.RandomRotation(40),
      torchvision.transforms.ToTensor(),
      torchvision.transforms.Normalize(
          mean=[0.4914, 0.4822, 0.4465], std=[0.2023, 0.1994, 0.2010]
      ),
    ],
```

```

    ]
)
database = ImageFolder(data_dir, transform=transformer)

```

In [7]:

```

xtest =
train_datagen.flow_from_directory('/content/drive/MyDrive/Classroom/Dataset
Plant Disease/Veg-dataset/Veg-dataset/test_set',
                                target_size=(64,64),
                                class_mode='categorical',
                                batch_size=10)

```

Found 0 images belonging to 9 classes.

In [8]:

```

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import
Convolution2D,MaxPooling2D,Flatten,Dense

```

4.Add CNN Layers

In [9]:

```

model = Sequential()
model.add(Convolution2D(32, (3, 3), activation='relu', input_shape=(64, 64, 3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(300, activation='relu'))
model.add(Dense(150, activation='relu'))
model.add(Dense(4, activation='softmax'))

```

5.Compile the model

In [10]:

```

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['ac
curacy'])

```

6.Fit the model

In [11]:

```

from keras.callbacks import EarlyStopping,ReduceLRonPlateau

```

In [12]:

```

early_stopping=EarlyStopping(monitor='val_accuracy',
                             patience=5)
reduce_lr=ReduceLRonPlateau(monitor='val_accuracy',
                             patience=5,
                             factor=0,min_lr=0.00001)
callback= [reduce_lr,early_stopping]

```

In [13]:

```

model.fit_generator(xtrain,
                    steps_per_epoch=len(xtrain),
                    epochs=100,
                    callbacks=callback,
                    validation_data=xtest,
                    validation_steps=len(xtest))

```

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: UserWarning
: `Model.fit_generator` is deprecated and will be removed in a future versi
on. Please use `Model.fit`, which supports generators.

Epoch 1/100

```

InvalidArgumentError                                Traceback (most recent call last)
in
      4             callbacks=callback,
      5             validation_data=xtest,
----> 6             validation_steps=len(xtest))

/usr/local/lib/python3.7/dist-packages/keras/engine/training.py in fit_generator(self, generator, steps_per_epoch, epochs, verbose, callbacks, validation_data, validation_steps, validation_freq, class_weight, max_queue_size, workers, use_multiprocessing, shuffle, initial_epoch)
    2272         use_multiprocessing=use_multiprocessing,
    2273         shuffle=shuffle,
-> 2274         initial_epoch=initial_epoch)
    2275
    2276 @doc_controls.do_not_generate_docs

/usr/local/lib/python3.7/dist-packages/keras/utils/traceback_utils.py in error_handler(*args, **kwargs)
    65     except Exception as e: # pylint: disable=broad-except
    66         filtered_tb = _process_traceback_frames(e.__traceback__)
--> 67         raise e.with_traceback(filtered_tb) from None
    68     finally:
    69         del filtered_tb

/usr/local/lib/python3.7/dist-packages/tensorflow/python/eager/execute.py in quick_execute(op_name, num_outputs, inputs, attrs, ctx, name)
    53     ctx.ensure_initialized()
    54     tensors = pywrap_tfe.TFE_Py_Execute(ctx._handle, device_name, op_name,
--> 55                                         inputs, attrs, num_outputs)
    56 except core._NotOkStatusException as e:
    57     if name is not None:

InvalidArgumentError: Graph execution error:

Detected at node 'categorical_crossentropy/softmax_cross_entropy_with_logits' defined at (most recent call last):
  File "/usr/lib/python3.7/runpy.py", line 193, in _run_module_as_main
    "__main__", mod_spec)
  File "/usr/lib/python3.7/runpy.py", line 85, in _run_code
    exec(code, run_globals)
  File "/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py", line 16, in <module>
    app.launch_new_instance()
  File "/usr/local/lib/python3.7/dist-packages/traitlets/config/application.py", line 846, in launch_instance
    app.start()
  File "/usr/local/lib/python3.7/dist-packages/ipykernel/kernelapp.py", line 612, in start
    self.io_loop.start()
  File "/usr/local/lib/python3.7/dist-packages/tornado/platform/asyncio.py", line 149, in start
    self.asyncio_loop.run_forever()
  File "/usr/lib/python3.7/asyncio/base_events.py", line 541, in run_forever
    self._run_once()

```

```

File "/usr/lib/python3.7/asyncio/base_events.py", line 1786, in _run_on
ce
    handle._run()
File "/usr/lib/python3.7/asyncio/events.py", line 88, in _run
    self._context.run(self._callback, *self._args)
File "/usr/local/lib/python3.7/dist-packages/tornado/ioloop.py", line 6
90, in
    lambda f: self._run_callback(functools.partial(callback, future))
File "/usr/local/lib/python3.7/dist-packages/tornado/ioloop.py", line 7
43, in _run_callback
    ret = callback()
File "/usr/local/lib/python3.7/dist-packages/tornado/gen.py", line 787,
in inner
    self.run()
File "/usr/local/lib/python3.7/dist-packages/tornado/gen.py", line 748,
in run
    yielded = self.gen.send(value)
File "/usr/local/lib/python3.7/dist-packages/ipykernel/kernelbase.py",
line 365, in process_one
    yield gen.maybe_future(dispatch(*args))
File "/usr/local/lib/python3.7/dist-packages/tornado/gen.py", line 209,
in wrapper
    yielded = next(result)
File "/usr/local/lib/python3.7/dist-packages/ipykernel/kernelbase.py",
line 268, in dispatch_shell
    yield gen.maybe_future(handler(stream, idents, msg))
File "/usr/local/lib/python3.7/dist-packages/tornado/gen.py", line 209,
in wrapper
    yielded = next(result)
File "/usr/local/lib/python3.7/dist-packages/ipykernel/kernelbase.py",
line 545, in execute_request
    user_expressions, allow_stdin,
File "/usr/local/lib/python3.7/dist-packages/tornado/gen.py", line 209,
in wrapper
    yielded = next(result)
File "/usr/local/lib/python3.7/dist-packages/ipykernel/ipkernel.py", li
ne 306, in do_execute
    res = shell.run_cell(code, store_history=store_history, silent=silent
)
File "/usr/local/lib/python3.7/dist-packages/ipykernel/zmqshell.py", li
ne 536, in run_cell
    return super(ZMQInteractiveShell, self).run_cell(*args, **kwargs)
File "/usr/local/lib/python3.7/dist-packages/IPython/core/interactivesh
ell.py", line 2855, in run_cell
    raw_cell, store_history, silent, shell_futures)
File "/usr/local/lib/python3.7/dist-packages/IPython/core/interactivesh
ell.py", line 2881, in _run_cell
    return runner(coro)
File "/usr/local/lib/python3.7/dist-packages/IPython/core/async_helpers
.py", line 68, in _pseudo_sync_runner
    coro.send(None)
File "/usr/local/lib/python3.7/dist-packages/IPython/core/interactivesh
ell.py", line 3058, in run_cell_async
    interactivity=interactivity, compiler=compiler, result=result)
File "/usr/local/lib/python3.7/dist-packages/IPython/core/interactivesh
ell.py", line 3249, in run_ast_nodes
    if (await self.run_code(code, result,  async_=asy)):

```

```

File "/usr/local/lib/python3.7/dist-packages/IPython/core/interactiveshell.py", line 3326, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
File "", line 6, in
    validation_steps=len(xtest))
File "/usr/local/lib/python3.7/dist-packages/keras/engine/training.py",
line 2274, in fit_generator
    initial_epoch=initial_epoch)
File "/usr/local/lib/python3.7/dist-packages/keras/utils/traceback_util
s.py", line 64, in error_handler
    return fn(*args, **kwargs)
File "/usr/local/lib/python3.7/dist-packages/keras/engine/training.py",
line 1409, in fit
    tmp_logs = self.train_function(iterator)
File "/usr/local/lib/python3.7/dist-packages/keras/engine/training.py",
line 1051, in train_function
    return step_function(self, iterator)
File "/usr/local/lib/python3.7/dist-packages/keras/engine/training.py",
line 1040, in step_function
    outputs = model.distribute_strategy.run(run_step, args=(data,))
File "/usr/local/lib/python3.7/dist-packages/keras/engine/training.py",
line 1030, in run_step
    outputs = model.train_step(data)
File "/usr/local/lib/python3.7/dist-packages/keras/engine/training.py",
line 890, in train_step
    loss = self.compute_loss(x, y, y_pred, sample_weight)
File "/usr/local/lib/python3.7/dist-packages/keras/engine/training.py",
line 949, in compute_loss
    y, y_pred, sample_weight, regularization_losses=self.losses)
File "/usr/local/lib/python3.7/dist-packages/keras/engine/compile_utils
.py", line 201, in __call__
    loss_value = loss_obj(y_t, y_p, sample_weight=sw)
File "/usr/local/lib/python3.7/dist-packages/keras/losses.py", line 139
, in __call__
    losses = call_fn(y_true, y_pred)
File "/usr/local/lib/python3.7/dist-packages/keras/losses.py", line 243
, in call
    return ag_fn(y_true, y_pred, **self._fn_kwargs)
File "/usr/local/lib/python3.7/dist-packages/keras/losses.py", line 178
8, in categorical_crossentropy
    y_true, y_pred, from_logits=from_logits, axis=axis)
File "/usr/local/lib/python3.7/dist-packages/keras/backend.py", line 51
35, in categorical_crossentropy
    labels=target, logits=output, axis=axis)
Node: 'categorical_crossentropy/softmax_cross_entropy_with_logits'
logits and labels must be broadcastable: logits_size=[10,4] labels_size=[10
,9]

```

```

[[[{{node categorical_crossentropy/softmax_cross_entropy_with_logits
s}}]] [Op:__inference_train_function_792]

```

7. Save the model

In [14]:

```
model.save('Veg.h5')
```

8. Test the model

In [15]:

```
import numpy as np
```

```
from tensorflow.keras.preprocessing import image
```

In [16]:

```
img = image.load_img('/content/drive/MyDrive/Classroom/Dataset Plant  
Disease/Veg-dataset/Veg-  
dataset/train_set/Pepper,_bell___Bacterial_spot/0022d6b7-d47c-4ee2-ae9a-  
392a53f48647___JR_B.Spot_8964.JPG',target_size=(94,94))
```

In [17]:

```
img
```

Out[17]:



In [18]:

```
x = image.img_to_array(img)  
x
```

Out[18]:

```
array([[113.,  98., 101.],  
       [143., 128., 131.],  
       [108.,  93.,  96.],  
       ...,  
       [165., 149., 150.],  
       [170., 154., 155.],  
       [163., 147., 148.]],  
  
       [[111.,  96.,  99.],  
       [131., 116., 119.],  
       [134., 119., 122.],  
       ...,  
       [162., 146., 147.],  
       [168., 152., 153.],  
       [167., 151., 152.]],  
  
       [[115., 100., 103.],  
       [129., 114., 117.],  
       [125., 110., 113.],  
       ...,  
       [173., 157., 158.],  
       [169., 153., 154.],  
       [174., 158., 159.]],  
  
       ...,  
  
       [[151., 138., 145.],  
       [151., 138., 145.],  
       [177., 164., 171.],  
       ...,  
       [169., 156., 165.],  
       [173., 160., 169.],  
       [164., 151., 160.]],  
  
       [[200., 187., 194.],  
       [202., 189., 196.]])
```

```

[158., 145., 152.],
...,
[167., 154., 163.],
[165., 152., 161.],
[171., 158., 167.]],

[[141., 128., 135.],
[172., 159., 166.],
[147., 134., 141.],
...,
[174., 161., 170.],
[169., 156., 165.],
[172., 159., 168.]]], dtype=float32)

```

In [19]:

```

x = np.expand_dims(x,axis=0)
x

```

Out[19]:

```

array([[[[113.,  98., 101.],
         [143., 128., 131.],
         [108.,  93.,  96.],
         ...,
         [165., 149., 150.],
         [170., 154., 155.],
         [163., 147., 148.]],

        [[111.,  96.,  99.],
         [131., 116., 119.],
         [134., 119., 122.],
         ...,
         [162., 146., 147.],
         [168., 152., 153.],
         [167., 151., 152.]],

        [[115., 100., 103.],
         [129., 114., 117.],
         [125., 110., 113.],
         ...,
         [173., 157., 158.],
         [169., 153., 154.],
         [174., 158., 159.]],

        ...,

        [[151., 138., 145.],
         [151., 138., 145.],
         [177., 164., 171.],
         ...,
         [169., 156., 165.],
         [173., 160., 169.],
         [164., 151., 160.]],

        [[200., 187., 194.],
         [202., 189., 196.],
         [158., 145., 152.],
         ...,
         [167., 154., 163.],

```

```

[165., 152., 161.],
[171., 158., 167.]],

[[141., 128., 135.],
[172., 159., 166.],
[147., 134., 141.],
...,
[174., 161., 170.],
[169., 156., 165.],
[172., 159., 168.]]]], dtype=float32)

```

In [20]:

```

from keras.callbacks import EarlyStopping, ReduceLROnPlateau

```

In [21]:

```

early_stopping = EarlyStopping(monitor='val_accuracy',
                                patience=5)
reduce_lr = ReduceLROnPlateau(monitor='val_accuracy',
                                patience=5,
                                factor=0.5,min_lr=0.00001)

callback = [reduce_lr,early_stopping]

```

In [22]:

```

img =
image.load_img('/content/drive/MyDrive/Classroom/flowers/dandelion/10486992
895_20b344ce2d_n.jpg',target_size=(64,64))
x = image.img_to_array(img)
x = np.expand_dims(x,axis=0)
pred = np.argmax(model.predict(x))

1/1 [=====] - 0s 105ms/step

```